

Figure 1
PRIOR ART

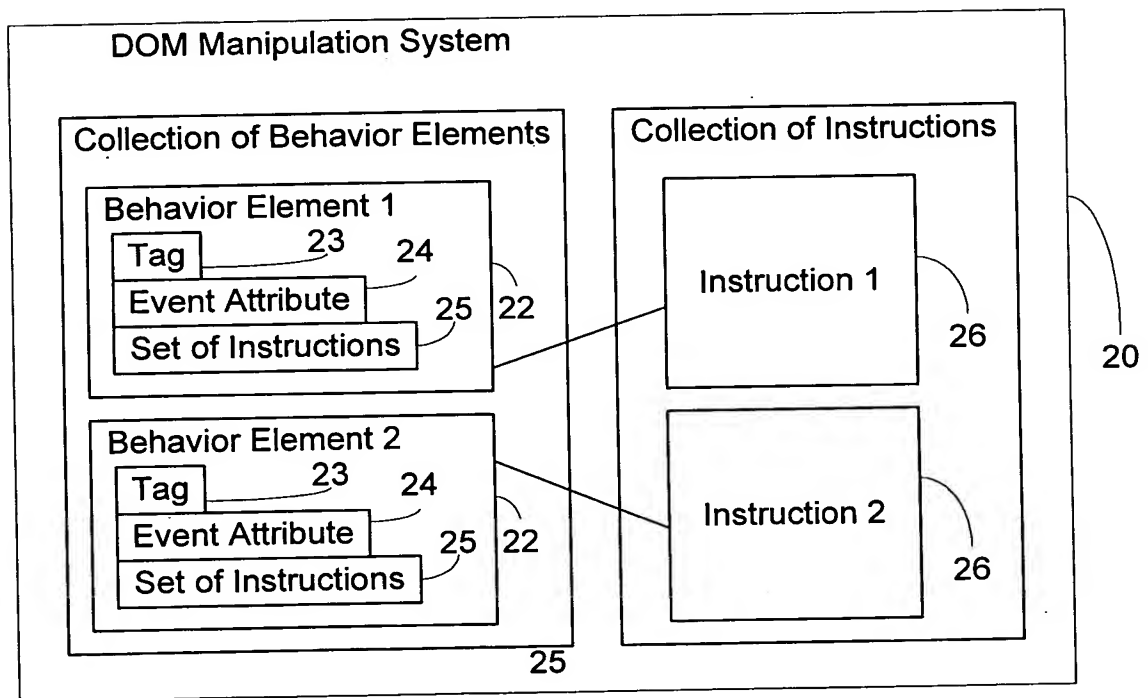


Figure 2

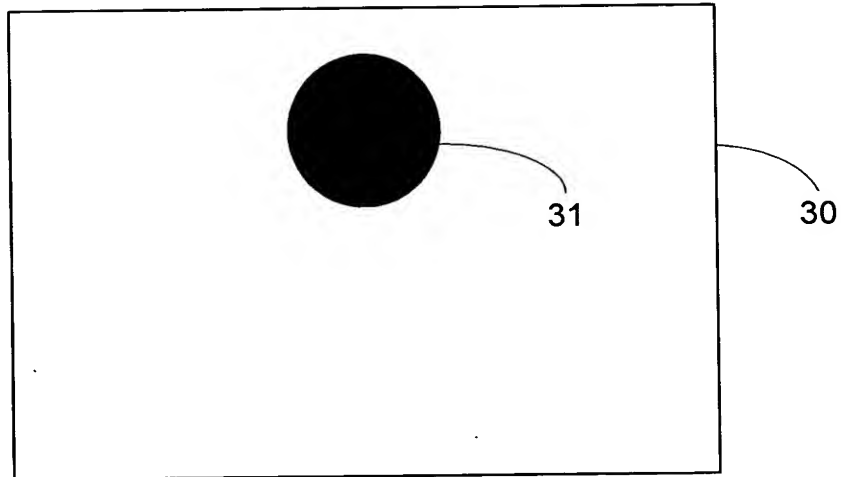


Figure 3

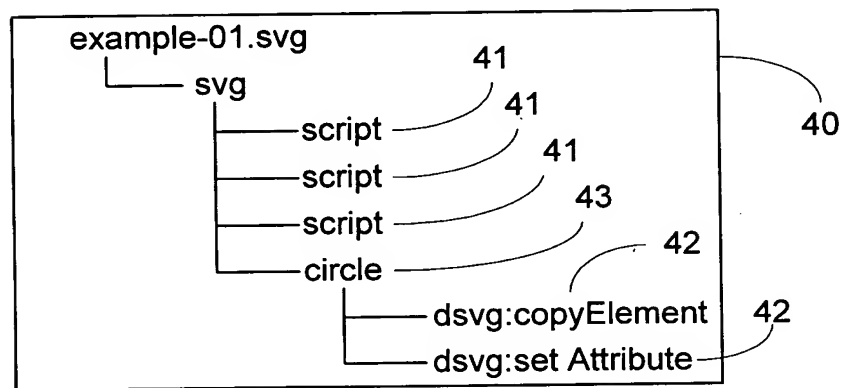


Figure 4

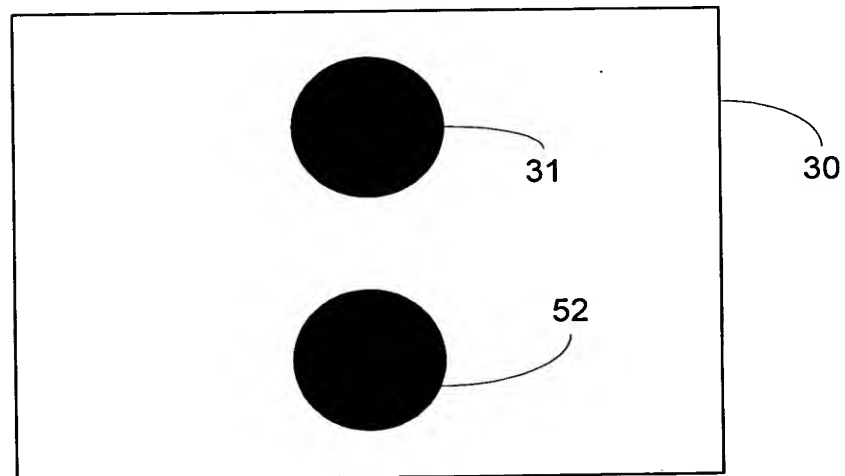


Figure 5

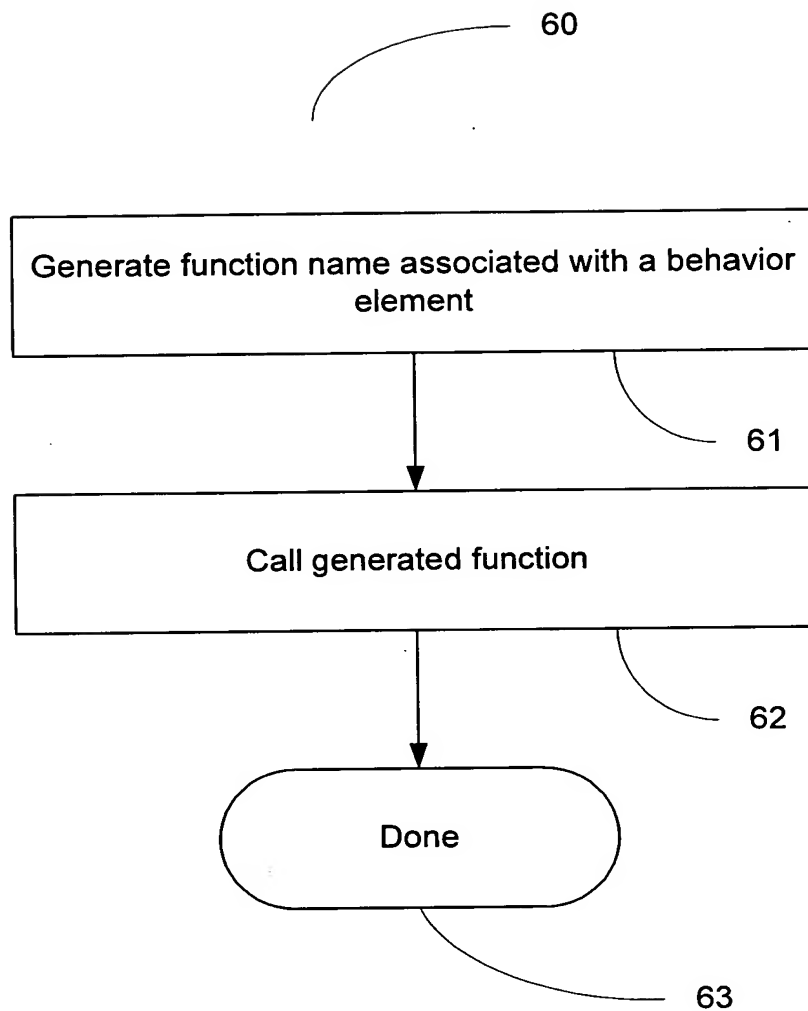


Figure 6

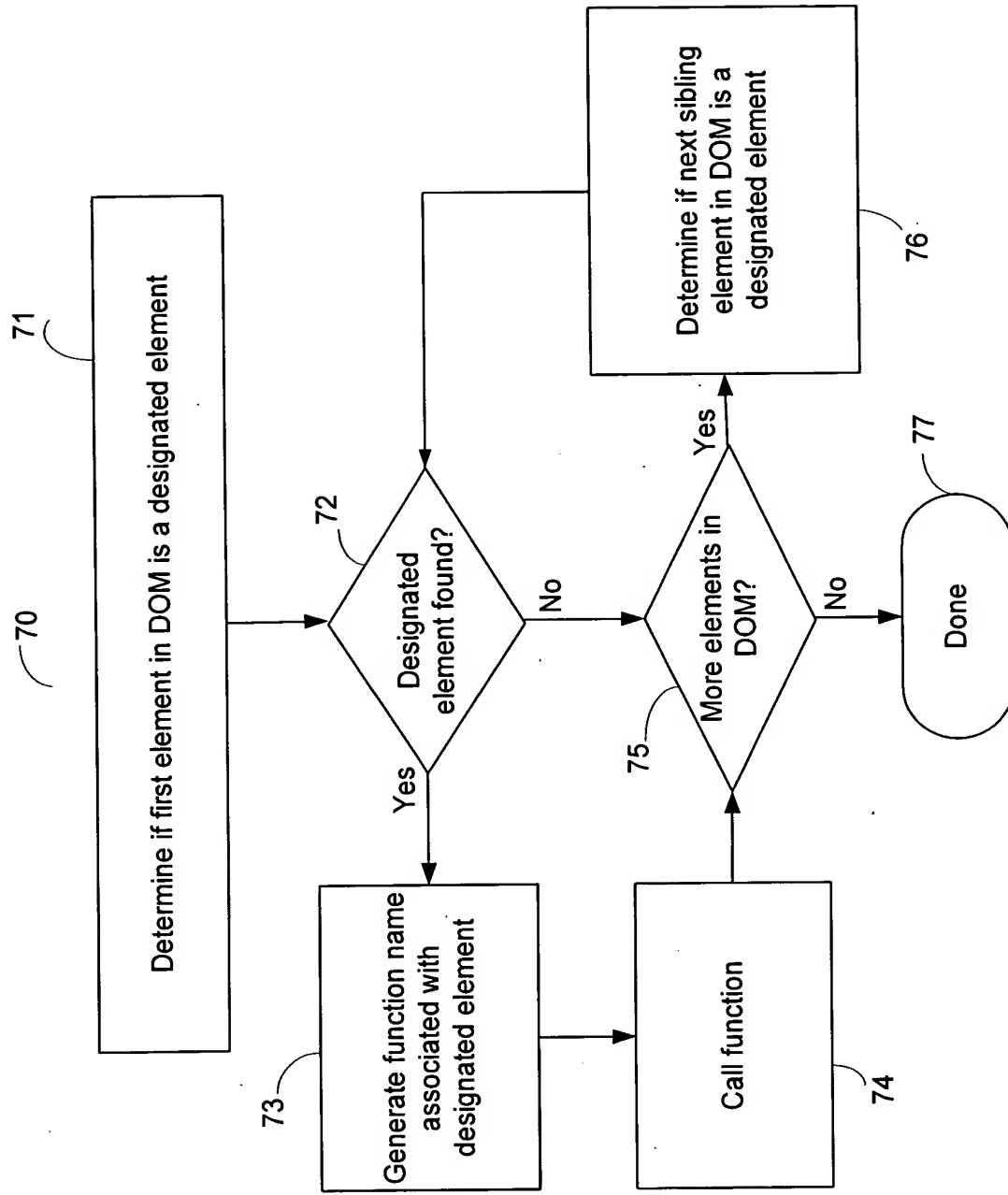


Figure 7

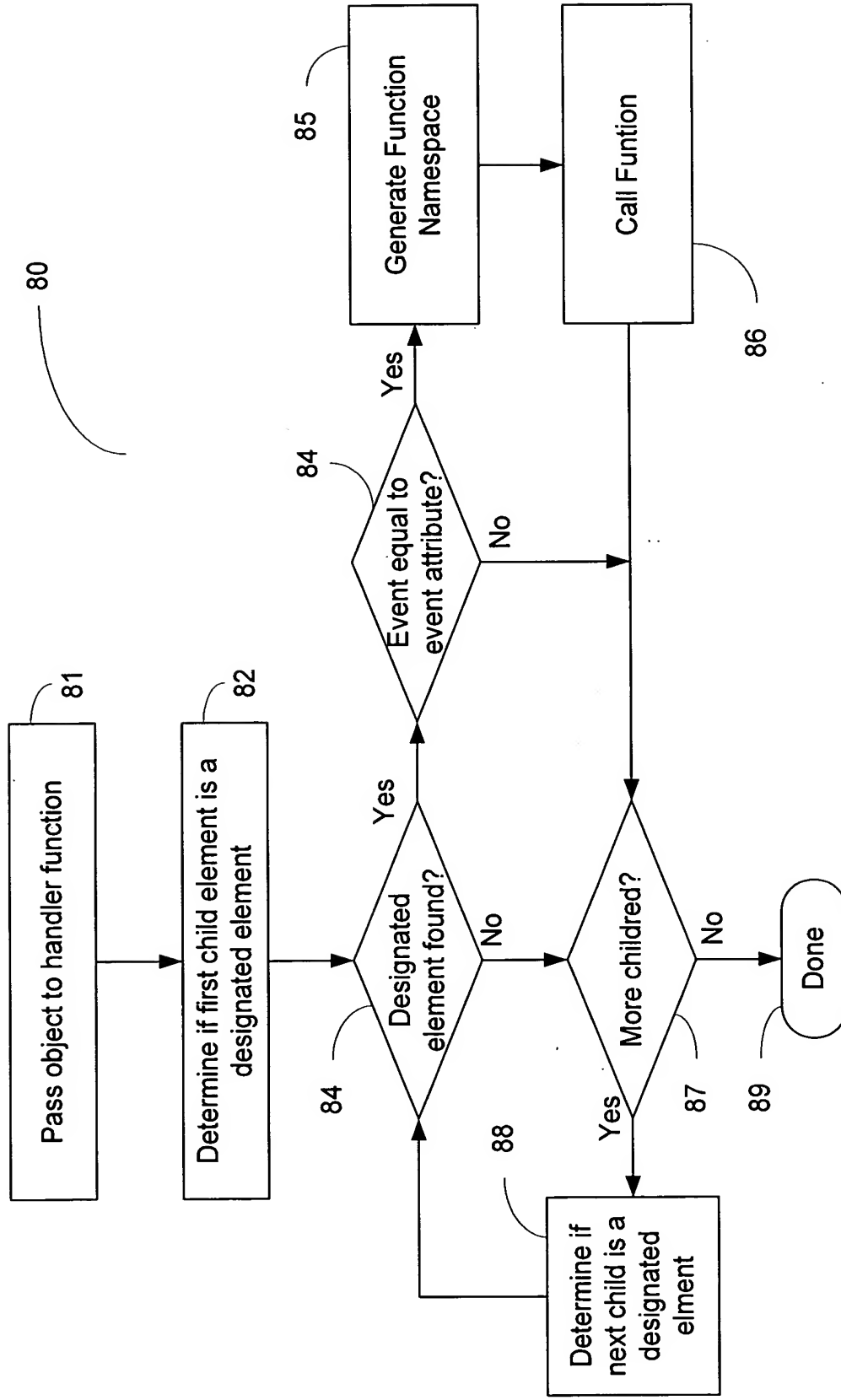


Figure 8

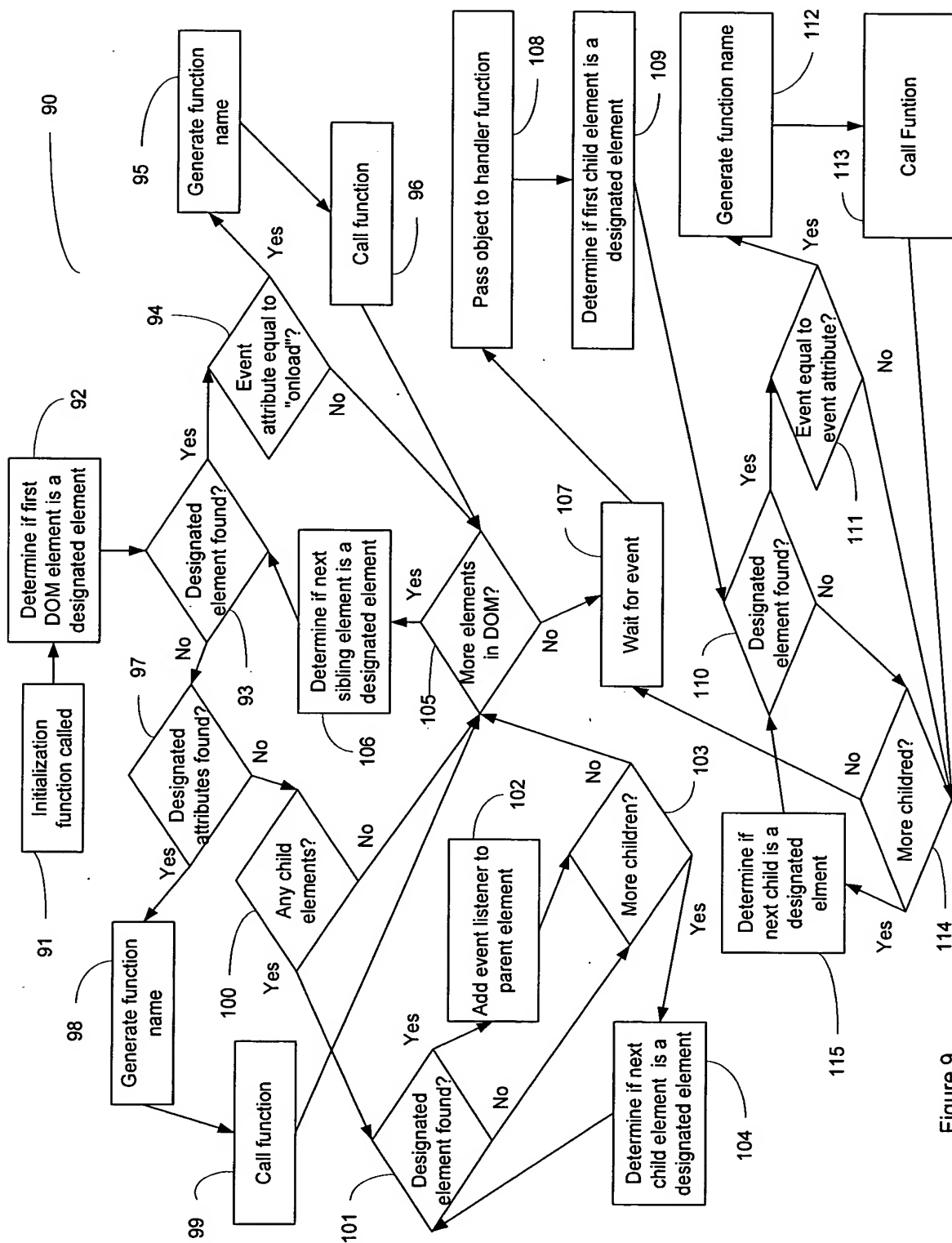


Figure 9

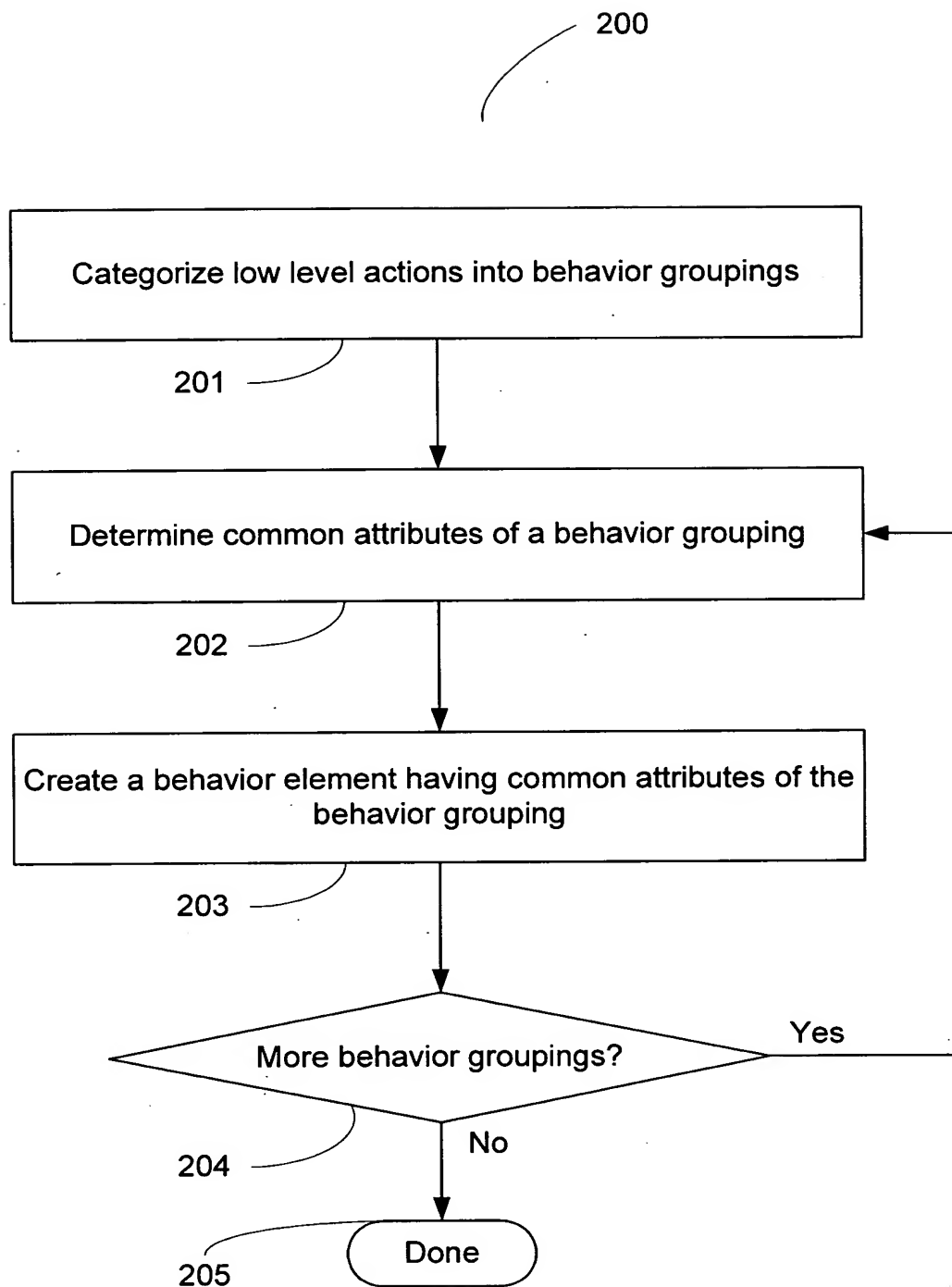


Figure 10

210

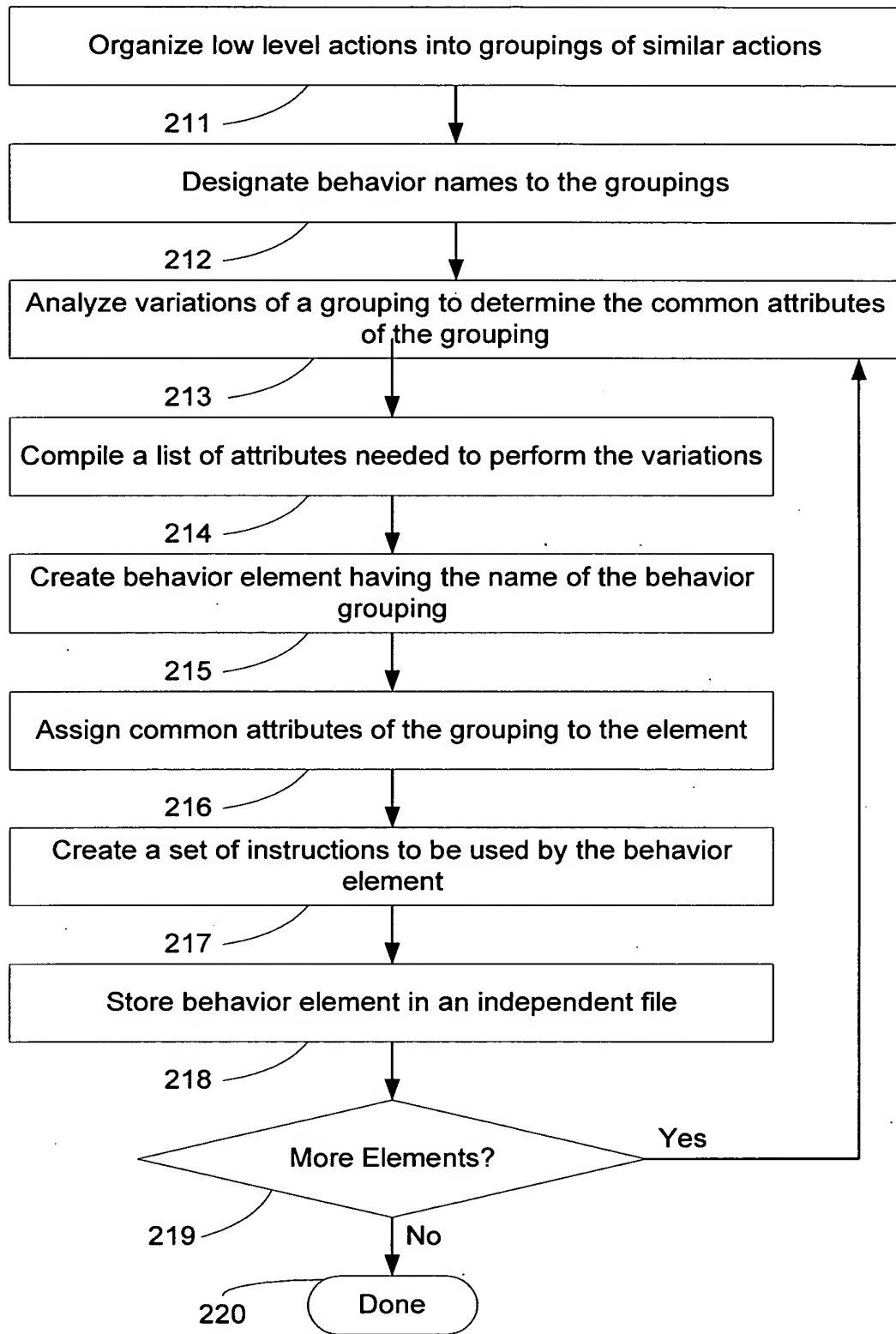
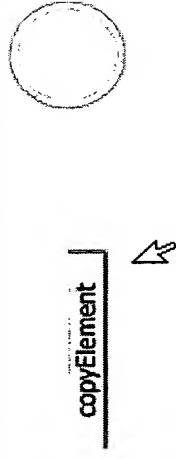
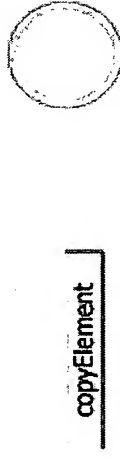


Figure 11

dSVG sample behavior: copyElement



Copy element used with sourceElementID mode of operation.



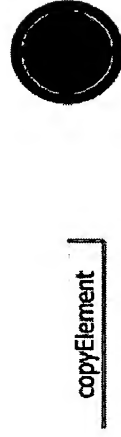
Copy element used in conjunction with setStyle.

Content of file: dsvg:copyElement

The dsvg:copyElement will copy the specified target element and generate a cloned element. Clicking the button will create a solid blue circle with a dark blue border over the transparent one.

Figure 12A

dSVG sample behavior: copyElement



Copy element used with sourceElementID mode of operation.



Copy element used in conjunction with setStyle.

Content of file: dsvg:copyElement

The dsvg:copyElement will copy the specified target element and generate a cloned element.
Clicking the button will create a solid blue circle with a dark blue border over the transparent one.

Figure 12B

dSVG sample behavior: createElement

Pressing the button will create a solid blue circle with a dark blue border over top of the transparent one.



Content of file: `dsvg:createElement`, `dsvg:setAttribute`
The `dsvg:createElement` will create a new solid circle over top of the transparent one when the button is selected.
The new element is inserted into the specified location within the DOM.

Figure 13A

dSVG sample behavior: createElement

Pressing the button will create a solid blue circle with a dark blue border over top of the transparent one.



Content of file: `dsvg.createElement`, `dsvg.setAttribute`
The `dsvg.createElement` will create a new solid circle over top of the transparent one when the button is selected.
The new element is inserted into the specified location within the DOM.

Figure 13B

dSVG sample behavior: findElements



findElements conditions are:

nodeID begins with a 't' and ends with an 'o'
nodeID begins with an 't' and ends with an 'e'
OR the element width attribute contains a '3'

Return find

Selecting the button will return the nodeID's that will be added to the outputList

Content of file: dsvg.findElements

The dsvg.findElements will find the NodeID's and return them to an output list.

The conditions specified can include the use of '*' wildcards when searching for IDs.

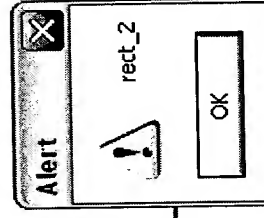



Figure 14

dSVG sample behavior: loadXML				
group 1	1. Basic loadXML samples		2. Synchronization of non-linear events	
group 2	<div>basic</div> <div>w/ If </div> <div>w/ loop</div>		<div>with loadXML</div> <div>without loadXML</div> <div>○</div>	
group 3			<p>This sample goes through a series of loops / conditional statements. The resulting location is returned to an alert.</p>	
group 4	<div>reset default</div>		<p>3. dsvg:loadXML using docID attribute.</p> <ul style="list-style-type: none"> - The docID attribute is intended for arbitrary XML - Allows access to data that resides in an outside fragment. 	
group 5	<p>Resulting load order</p> <p>.....</p>		<div>load DocID</div> <p>the cx value from fragment (fragment.svg#ellipse1) will be placed here.</p>	

Section 1 illustrates basic usage of dsvg:loadXML.

Section 2 illustrates how loadXML can be used synchronously with non-linear events.

Section 3 illustrates how the 'docID' attribute can be used to retrieve data from outside document fragments.

Figure 15A

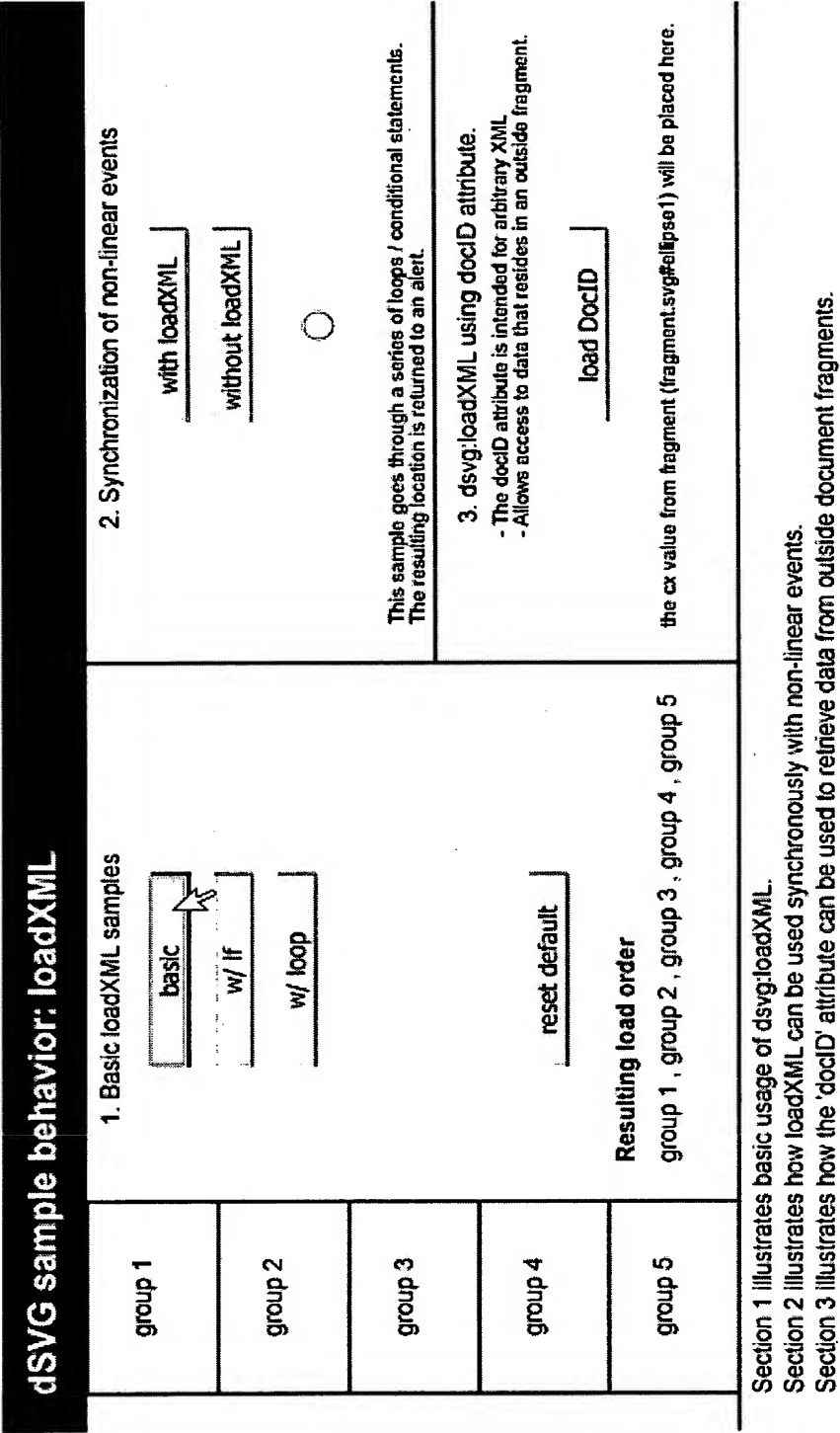


Figure 15B

dSVG sample behavior: moveElement

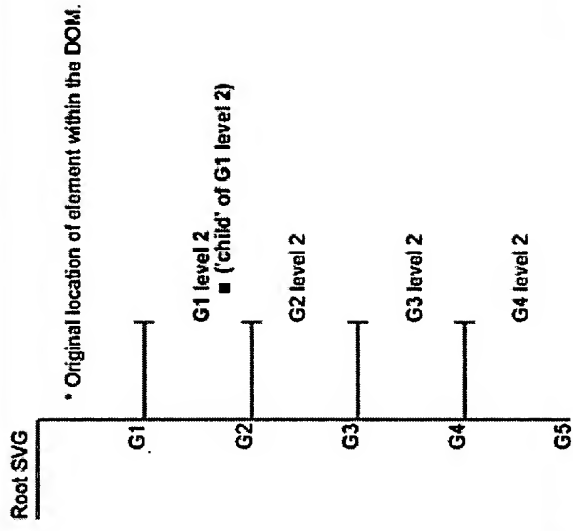
Select radio buttons to move element within the DOM.

☒ moveElement: child

☐ moveElement: sibling

☐ moveElement: parent

☐ moveElement: replacement



*Note: The chart on the right illustrates the corresponding DOM.

Content of file: dsvg.moveElement

The dsvg.moveElement will move the source element to a specified target location within the DOM.
The rectangle within the chart will track the location where the element is being inserted.

Figure 16

dSVG sample behavior: setAttribute

setAttribute



Content of file: dsvg:setAttribute
The dsvg:setAttribute element will set the attributes of the specified target element.

Figure 17A

dSVG sample behavior: setAttribute

setAttribute



Content of file: dsvg:setAttribute
The dsvg:setAttribute element will set the attributes of the specified target element.

Figure 17B

dSVG sample behavior: setData

setData	Label
---------	-------

Content of file: dsvg:setData
The dsvg:setData element will set a text node with the specified data.

Figure 18A

dSVG sample behavior: setData

setData


This is a sample of setData.

Content of file: dsvg:setData
The dsvg:setData element will set a text node with the specified data.

Figure 18B

dSVG sample behavior: setStyle

setStyle



Content of file: dsvg.setStyle
The dsvg.setStyle element will set the style of a specified target element.

Figure 19A

dSVG sample behavior: setStyle



Content of file: dsvg.setStyle
The dsvg.setStyle element will set the style of a specified target element.

Figure 19B

dSVG sample behavior: setTransform

setTransform



Content of file: dsvg:setTransform
The dsvg:setTransform element will transform the specified target element.

Figure 20A

dSVG sample behavior: setTransform

setTransform



Content of file: dsvg:setTransform
The dsvg:setTransform element will transform the specified target element

Figure 20B